Modeling Model-View-Controller (MVC) Architecture Pattern as System of Systems (SoS) to Aid Comprehension and Reasoning

Mrityunjay Kumar
IIIT Hyderabad
India
mrityunjay.k@research.iiit.ac.in

Venkatesh Choppella
IIIT Hyderabad
India
venkatesh.choppella@iiit.ac.in

ABSTRACT

As software systems become increasingly complex, it is important that students develop *software-is-system* mindset. Our research aims to help engineers comprehend architecture patterns better using systems approach. We postulate that such an understanding is aided when we enhance pattern descriptions with a System of Systems (SoS) model of the pattern to better explicate the dynamics. We demonstrate this for Model-View-Controller (MVC) pattern, a common architecture pattern for interactive applications. Such an SoS model of the pattern can also be beneficial in the teaching-learning process in a software engineering classroom.

CCS CONCEPTS

• **Computing methodologies** → **Modeling methodologies**.

KEYWORDS

architecture pattern, transition systems, systems modeling

ACM Reference Format:
Mrityunjay Kumar and Venkatesh Choppella. 2023. Modeling Model-View-Controller (MVC) Architecture Pattern as System of Systems (SoS) to Aid Comprehension and Reasoning. In *Proceedings of the ACM Conference on Global Computing Education Vol 2 (CompEd 2023), December 5–9, 2023, Hyderabad, India.* ACM, New York, NY, USA, 1 page. https://doi.org/10.1145/3617650.3624934

1 INTRODUCTION

Existing architecture pattern descriptions use UML diagrams (which lack precision) and code fragments (which include low level details). We believe that the dynamics of the pattern can be better captured and understood if we can model it as System of Systems (SoS) and use systems vocabulary to do so.

A **communicating system** S can be written as a Transition System (a variation on its use in [1]) $S = \{X, X^0, f, P\}$ where X is the state space, X^0 is the initial state space, f is the transition function, and P is the set of ports (both input and output). A port is an abstract entity through which a system can communicate with another system.

A **system of systems** SoS can be written as $SoS = (S, C, B, R)$ where S is the set of component systems, C is the set of channels (an abstract entity that represents connection between two ports for unidirectional communication), B is the behavior of SoS (captured as the set of possible sequence of message exchanges in the system) and R is the set of constraints on SoS behavior (in terms of metrics of the traces). We use Communicating Sequential Processes (CSP) notation for B and R.

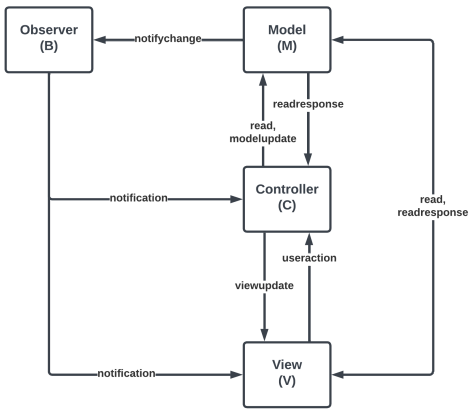

Figure 1: MVC SoS

We use the above definitions to model the MVC pattern as an SoS.

S consists of 4 systems: V (the view system), C (the controller system), M (the model system), and B (the observer system). **C** consists of 9 channels, see Fig 1. **B** consists of all possible traces of the *SoS*. For example, a useraction triggers a modelupdate message, which is followed by notifychange and notification messages, which in turn triggers read and readresponse. **R** consists of all significant constraints on the behavior. For example, a constraint can be: every trace has a useraction as the beginning. Another could be: every trace has either a viewupdate or modelupdate as the second message.

By breaking down the dynamics into these distinct components and using a systems vocabulary to describe SoS and the four communicating systems, we make the MVC pattern more amenable to comprehension and reasoning as will be demonstrated in the poster.

REFERENCES

[1] Venkatesh Choppella, Kasturi Viswanath, and Mrityunjay Kumar. 2021. Algodynamics: algorithms as systems. In *2021 IEEE Frontiers in Education Conference (FIE)*. IEEE, 1–9.

CompEd 2023, December 5–9, 2023, Hyderabad, India

ACM ISBN 979-8-4007-0374-4/23/12.
https://doi.org/10.1145/3617650.3624934